



## C1 - 3 数据类型

### // 学习目标

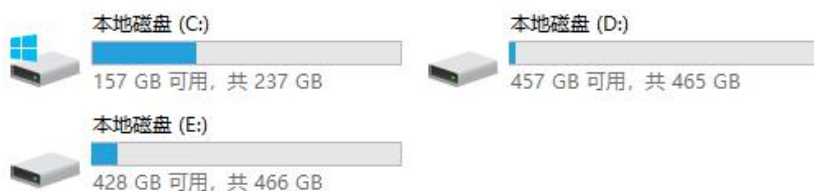
- 认识基本的数据类型
  - 掌握数据类型的转换方式
  - 了解 ASCII 码
- 

### // 思考：计算机是如何存储数据的呢？



原理上是把各种数据转换成 0 或 1 来进行存储，也就是二进制数。

我们在计算机中打字或者下载一些资料，都会使用到计算机的存储空间（简称 内存）。计算机的本地磁盘里已经预备好了大大的存储空间，先来看我的。



在这里再与大家分享一下数据之间的单位转换：

$$1 \text{ TB} = 1024 \text{ GB}$$

$$1 \text{ GB} = 1024 \text{ MB}$$

$$1 \text{ MB} = 1024 \text{ KB}$$

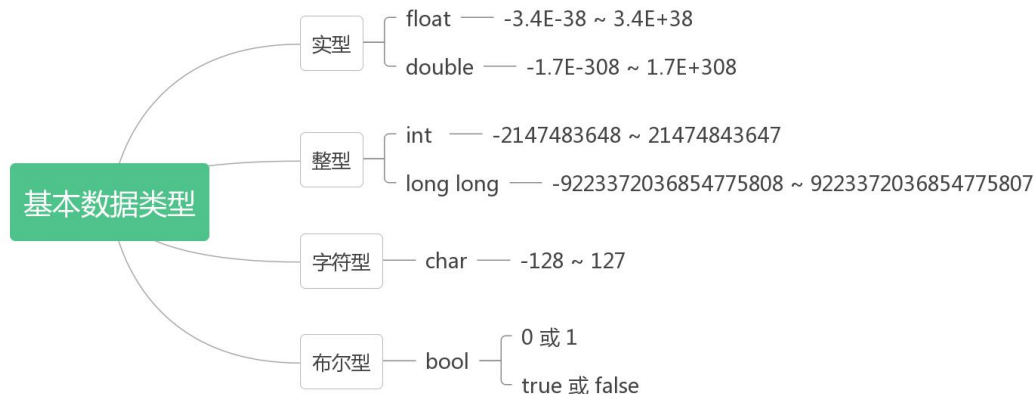
$$1 \text{ KB} = 1024 \text{ B}$$

$$1 \text{ B} = 1 \text{ 字节} = 8 \text{ bit (位)}$$

## // 数据类型分类

C++语言提供了丰富的数据类型：整型、实型、字符型、布尔型。它们都是系统定义的简单数据类型，称为标准数据类型。

以下是几种基本数据类型，后面的数字表示该类型的数据范围。



**\*注意：**这里的实型是分为单精度浮点型（float）和双精度浮点型（double）单精度浮点型的精度和有效位数都比双精度浮点型的低，一般单精度浮点型可以保留 **7** 位有效数位（有效数位是能显示出来的，并且是从左往右第一个不为零的数字开始数，比如我们在计算之后得到的结果是 003456 这样的整数，可是结果却只能显示成 3456，前面的 0 会自动被去掉，所以它的有效位数只有 4 位），而双精度浮点型的有效位数可达 **15** 位。

```
5 int main() {
6     float a = 3.141592612;
7     double b = 3.141592612;
8     cout << setprecision(10) << a << endl;
9     cout << setprecision(10) << b;
10    return 0;
11 }
12
```

3.141592503  
3.141592612  
Process exited after 0.1126 seconds with return code 0

这段代码即可说明两种类型的优缺点，当我们都想保留 10 位的精度时，float 只能保留前 7 位，7 位之后的数字已经不再准确，而 double 却保留的很完整！

## // 如何使用不同种的数据类型呢？

和之前我们定义变量一样，大家都用习惯了 `int` 这个关键词，现在我们来试试所有的数据类型吧！

```
int a = 123;

double b = 3.1415926;

char c = 'A' ;

bool d = true;
```

这里写完之后再写一个 `cout` 语句把它们都输出出来，看看结果是什么呢？

如果你有其他有想法的话可以改一些地方，多多探索吧！

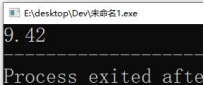
---

## // 数据类型之间的转换

不同类型的数据放在一起计算就会发生类型转换，有自动转换和强制转换。


**自动转换：**小遇大变大。比如 `int` 类型和 `double` 类型的数据一起计算后得到的结果就是 `double` 类型，这里就是只保留了更大的数据类型！

```
int main() {
    int a = 3;
    double b = 3.14;
    cout << a*b;
    return 0;
}
```



**强制转换：**把一个数据赋值给另一个和自己不同类型的数据里，就会发生类型转换。

```
int a ;
double b = 1.98;
a = b; // 这里是强制转换
cout << a << endl << b;
return 0;
}
```



强制转换还有另外一种方式，格式如下：

(需要转换的类型) + 要转换的数据

// 这里是把整数转换为字符

```
int main() {
    int a = 65;
    cout << (char) a;
    return 0;
}
```



思考：整数转换为字符是怎么回事？依据什么进行转换的？

转换 A 这个字符为整数就会得到 65，这里是依据 ASCII 码进行转换的。

// ASCII 码表

序号	字符	序号	字符	序号	字符	序号	字符	序号	字符	序号	字符
32	空格	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	\$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	deL

ASCII 码表里记录了每一个字符对应的序号，通过观察你会发现，字符转换

为整数之后不就是它所对应的序号嘛~ 🤔

## // 数据类型简单应用

### 【与圆相关的计算】

给出圆的半径，求圆的直径、周长和面积。输入圆的半径实数  $r$ ，输出圆的直径、周长、面积，每个数保留小数点后 4 位。 $\pi$  取 3.1415926；

#### 输入

输入包含一个实数  $r$  ( $0 < r \leq 10000$ )，表示圆的半径。

#### 输出

输出一行，包含三个数，分别表示圆的直径、周长、面积，数与数之间以一个空格分开，每个数保留小数点后 4 位。

#### 输入样例

3.0

#### 输出样例

6.0000 18.8496 28.2743

解析：此类题需要先分析题目中所包含的数据类型，这道题中包含的  $\pi$  取 3.1415926，因此我们需要使用 `double` 类型来计算才能把最终结果完整保留！

代码如下：

```
#include <iostream>
#include <iomanip>
using namespace std ;
int main(){
    double pi = 3.1415926 , r , d , l , s ;
    cin >> r ;
    d = 2 * r ;
    l = pi * d ;
    s = pi * r * r ;
    cout << fixed << setprecision (4) << d << l << s ;
    return 0 ;
}
```